TCN

0. Abstract

- 오디오 합성 및 기계 번역과 같은 작업에서 합성곱 아키텍처가 순환신경망보다 더 뛰어난 성능을 보일 수 있음이 밝혀짐
- 간단한 합성곱 아키텍처가 LSTM과 같은 대표적인 순환 신경망보다 더 다양한 작업과 데이터셋에서 우수한 성능을 보임

1. Introduction

- 시퀀스 모델링과 순환신경망이 밀접하게 연관되어 있다는 일반적인 인식
- 최근 연구에서는 특정 합성곱 아키텍처가 오디오 합성, 단어 수준의 언어 모델링, 기계 번역과 같은 작업에서 SOTA를 달성할 수 있음이 밝혀짐
- 시퀀스 모델링이 특정 분야에 국한되어 성공적인 성능이 나오는 것인지 혹은 시퀀스 데이터 = RNN이 필수"라는 기존의 통념이 여전히 유효한 문제인지 재고해봐야함
- 순환 신경망이 강세를 보이는 전통적인 영역(“home turf”)=음성 모델링, 언어 모델링 등에서 합성곱과 순환 접근 방식을 직접 비교하도록 설계
- 모든 작업에 적용할 수 있는 **일반적인 시계열 합성곱 네트워크(Temporal Convolutional Network, TCN)** 아키텍처를 설계
- TCN은 위에서 언급한 최근 연구에 제안된 개념들을 반영하되, 단순하게 유지
- TCN 아키텍처는 LSTM 및 GRU와 같은 대표적인 순환 신경망보다 더 높은 정확도를 보일 뿐만 아니라, 더 단순하고 명확한 구조를 가지고 있음

2. Background

- CNN은 1980~90년대부터 음성 인식에 사용되었으며, 이후 NLP 작업(품사 태깅, 의미 역할 부여, 문장 및 문서 분류)에도 적용
- CNN은 최근에 기계 번역, 오디오 합성, 언어 모델링에서도 성과를 보임
- RNN은 과거 정보를 유지하며 시퀀스를 처리하는 모델로, 언어 모델링과 기계 번역에서 큰 성공을 거둠.
- 기본 RNN은 학습이 어려워 LSTM, GRU 같은 변형 모델이 널리 사용됨.
- 논문에서는 CNN과 RNN의 성능을 비교하여 각각의 강점을 분석하는 것을 목표로 함.

3. Temporal Convolutional Networks

시퀀스 예측을 위해 컨볼루션 네트워크 설계의 최적 사례를 간결한 기본 아키텍처로 정리
= 시계열 컨볼루션 네트워크(TCN), 하나의 아키텍처 계열을 설명하는 단순한 용어로 사용

- Causal 컨볼루션 사용
  -> 미래의 정보가 과거로 누출되지 않도록 보장
- 입력 시퀀스를 동일한 길이의 출력 시퀀스로 변환 가능
  -> 이는 RNN과 동일한 특성
- 매우 긴 과거 정보를 활용 가능
  -> 깊은 네트워크(residual Layer 포함)와 확장된 컨볼루션을 결합하여 효과적인 히스토리 크기를 크게 확

장

3.1 Sequence Modeling

- 시퀀스 모델링 작업의 목표는 입력 시퀀스 x_0 , ... , x_T 가 있을 때, 각 시점 ttt 에서 대응되는 출력 y_0 , ... , y_T 를 예측하는 것
- 핵심 제약 조건은 출력 y_t를 예측할 때, 과거의 입력 x_0 , ... , x_T 만 사용할 수 있어야 한다는 점
- 시퀀스 모델링 네트워크는 다음과 같은 함수 $\hat{y}_0, \ldots, \hat{y}_T = f(x_0, \ldots, x_T)$
- 이때, 함수 f 가 인과성 제약(causal constraint) 을 만족해야 함

3.2 Casual Convolutions

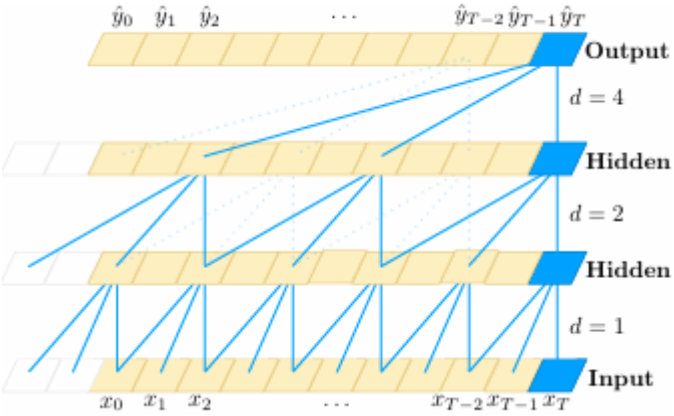
- 네트워크가 입력과 동일한 길이의 출력을 생성하기 위해 TCN은 1D FCN 아키텍처를 사용하며, 각 은닉 계층의 길이는 입력 계층과 동일하게 유지함
- 또, (커널 크기 - 1)만큼의 제로 패딩(zero padding)을 추가하여 이후 계층들의 길이가 이전 계층들과 동일하게 유지되도록 함
- 미래의 정보가 과거로 누출되지 않기 위해서 TCN은 인과적 합성곱을 사용하여, t 시점의 출력이 이전 계층에서 t 시점 및 그 이전 시점의 요소들과만 합성곱을 수행함
  $\text{TCN} = \text{1D FCN} + \text{causal convolutions.}$
- 이러한 기본적인 설계 방식의 주요 단점은 긴 과거 정보를 효과적으로 반영하기 위해 매우 깊은 네트워크를 구성하거나 매우 큰 필터를 사용해야한다는 것

3.3. Dilated Convolutions

- 단순한 인과적 합성곱은 네트워크의 깊이에 비례하는 크기의 과거 정보만을 참고할 수 있고, 시퀀스 작업에서 인과적 합성곱을 적용하는 데 어려움
- 위의 단점을 해결하기 위해 확장된 합성곱을 적용하여 넓은 수용 영역을 확보함

$$F(s) = (\mathbf{x} *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot \mathbf{x}_{s-d \cdot i}$$

  - d는 확장 계수(dilation factor)
  - k는 필터 크기(filter size)
  -$x_{s-d \cdot i}$는 과거 방향으로의 입력 값
- 확장 계수 d는 필터 탭 사이에 일정한 간격을 삽입하는 역할을 함
  -> d=1이라면, 확장된 합성곱은 일반적인 합성곱과 동일
  -> d값을 증가시키면 네트워크의 최상위 출력이 훨씬 넓은 범위의 입력을 반영할 수 있음
- TCN에서 수용 영역을 넓히기 위한 두 가지 방법
  1. 필터 크기 k를 증가시키는 방법
  2. 확장 계수 d를 증가시키는 방법
- 확장된 합성곱을 사용할 때는 네트워크 깊이에 따라 d 값을 지수적으로 증가시키는 방식이 흔히 사용됨
  -> 네트워크의 i번째 계층에서는 $d = O(2^i)$ 로 설정
- 위 방법을 통해 효과적인 수용 영역을 확보하면서 깊은 네트워크 구조에서도 입력 정보가 고르게 반영될 수 있음

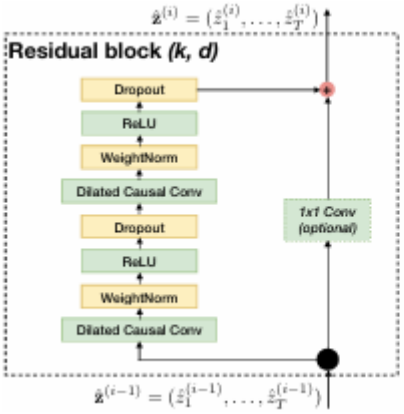


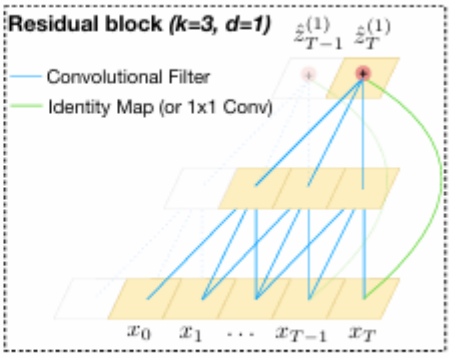
3.4. Residual Connections

- 잔차 블록에 쓰이는 연산은

  $$o = \text{Activation}(\mathbf{x} + \mathcal{F}(\mathbf{x}))$$

  - x는 블록의 입력
  - F(x)는 일련의 변환을 수행한 출력
  - o는 최종 출력
- 잔차 블록을 사용하면 네트워크가 항등 매핑에 대한 수정 사항을 학습하도록 유도
  -> 매우 깊은 네트워크에서 학습을 안정적으로 수행하는 데 유리
- TCN 네트워크를 깊고 크게 설계하는 과정에서 학습 안정성이 중요한 요소
- 따라서, TCN 기본 모델 설계할 때, 단순 합성곱 계층 대신 잔차 모듈을 사용
- 두 개의 확장된 인과적 합성곱 계층, ReLU 활성화 함수, 가중치 정규화를 통해 필터 학습 안정화, 각 합성곱 계층 뒤에 공간적 드롭아웃 추가하여 특정 채널을 완전히 제거





3.5. Discussion

TCN 모델 사용의 장점

- Paralleism
  RNN에서는 이후 시점의 예측이 이전 시점의 계산이 완료될 때까지 대기해야 하지만, 컨볼루션은 각 레이어에서 동일한 필터를 사용하기 때문에 병렬로 처리 가능 -> TCN 훈련 및 평가 시 긴 입력시퀀스를 한 번에 처리할 수 있음
- Flexible receptive field size
  더 많은 확장 컨볼루션 레이어를 쌓거나, 더 큰 확장 계수를 사용하거나, 필터 크기를 증가시키는 등의 방법으로 수용 영역 크기를 조정 -> 모델의 메모리를 잘 제어, 다양한 도메인에 쉽게 적응

- Stable gradients
  TCN은 시퀀스의 시간적 방향과 다른 역전파 경로를 가짐
- Low memory requirement for training
  TCN에서는 레이어 전체에서 필터가 공유되며, 역전파 경로는 네트워크 깊이에만 의존 -> 게이트가 있는 RNN이 TCN보다 많은 메모리를 사용할 가능성
- Variable length inputs
  TCN은 1D 컨볼루션 커널을 슬라이딩 방식을 사용하여 임의의 길이를 가진 시계열 데이터에 대해 RNN을 대체하여 사용할 수 있음

TCN 모델 사용의 단점

- Data storage during evaluation
  TCN은 유효 이력 길이만큼의 원본 시퀀스를 입력으로 받아야 하므로, 평가 시 더 많은 메모리를 사용
- Potential parameter change for a transfer of domain
  메모리 요구량이 작은 도메인에서 훈련된 모델을 더 긴 메모리가 필요한 도메인으로 이전할 경우, 충분한 수용 영역을 확보하지 못해 성능이 저하

5. Experiments

- RNN 모델의 홈구장같은 곳에서(주로 RNN이 실험되는 benchmark) TCN과 RNN의 성능을 비교 실험. 여러 도메인의 real data와 synthetic data를 포괄적으로 활용
- 동일한 TCN 아키텍처를 사용하되 network depth n, kernel size k를 달리하여 여러가지 버전으로 실험 exponential dilation d=2i (i: network depth)
- Gating mechanism, skip connection과 같은 정교한 아키텍쳐 작업은 RNN, TCN 모두 추가되지 않음. (vanilla model)

6. Conclusion

- dilation 및 residual connection과 같은 모범 사례를 causal convolution과 결합한 TCN 제안
- TCN 모델이 LSTM 및 GRU와 같은 일반적인 순환신경망보다 우수한 성능을 보임을 확인
- 장기 정보 전달에서 이론적으로 RNN은 "무한 기억"을 가질 수 있어야 하지만, 실제로는 기울기 소실 문제 등으로 인해 잘 발현되지 않음
- TCN은 딜레이션 합성곱과 잔차 연결을 활용하여 RNN보다 더 긴 기억을 유지할 수 있음